

Aufgabe: Alle Tabellenblätter Namens Tabelle1 aus allen Exceldateien eines Ordners einlesen und zusammenführen.

Ausgangslage: Im Ordner

OneDrive > Ralf – Persönlich > Dokumente > All_PQ > Dateien > Exceldateien > Einheitlich

Name	Status	Änderungsdatum	Typ	Größe
Elektroartikel.xlsx	✓	04.07.2025 10:15	Microsoft Excel-Ar...	13 KB
Haushaltswaren.xlsx	✓	04.07.2025 10:18	Microsoft Excel-Ar...	14 KB
Lebensmittel.xlsx	✓	27.12.2022 17:46	Microsoft Excel-Ar...	13 KB
Visualisierte Schrittfolge.docx	↻	05.07.2025 12:52	Microsoft Word-D...	514 KB

...befinden sich 3 Exceldateien mit gleichartiger Struktur, sowie ein Worddokument:

Die Exceldateien:

Elektroartikel.xlsx

Automatisches Speichern ✓ Elektroartikel.xlsx • Zuletzt

Datei Start Einfügen Zeichnen Seitenlayout Formeln Daten Überprüfen

Einfügen F K U A⁺ A⁻ Ausrichtung Zahl Bedingte Form

Zwischenablage Schriftart Formatvorlagen

ID	Produkt	Menge	Einheit	Preis	Bestand
1	Lampen	1	Stück	99	10
2	Stecker	1	Stück	0,8	12
3	Steckdosen	10	Stück	0,5	20
4	kabel 1,5	1	m	1,05	100

Tabelle1

Haushaltswaren.xlsx

Automatisches Speichern ✓ Haushaltswaren.xlsx

Datei Einfügen Zeichnen Seitenlayout Formeln Daten Überprüfen Ans

Tabellenname: tblHaushaltswaren Mit PivotTable zusammenfassen

Duplikate entfernen Datenschnitt einfügen Exportie

Tabellengröße ändern In Bereich konvertieren

Eigenschaften Tools Ext

ID	Produkt	Menge	Einheit	Preis	Bestand
1	Besen	1	Stück	99	10
2	Schaufel	1	Stück	0,8	12
3	Staublappen	10	Stück	0,5	20
4	Wäscheleine	1	m	1,05	100

Tabelle1

Lebensmittel.xlsx

Automatisches Speichern ✓ Lebensmittel... • Gespeichert

Datei Start Einfügen Zeichnen Seitenlayout Formeln Daten Überprüfen Ansicht

Einfügen F K U A⁺ A⁻ Ausrichtung Zahl Bedingte Formatierung

Zwischenablage Schriftart Formatvorlagen

ID	Produkt	Menge	Einheit	Preis	Bestand
1	Mehl	1	kg	0,99	10
2	Zucker	1	kg	0,8	12
3	Salz	500	g	0,5	20
4	Milch	1	l	1,05	5
5	Butter	1	Stück	1,5	20

Tabelle1

Allen gemeinsam ist, dass sich alle Daten im Arbeitsblatt (Sheet) **Tabelle1** befinden. Auch, wenn sich im Workbook **Haushaltswaren.xlsx** die Daten in einer *formatierten Tabelle* Namens **tblHaushaltswaren** befinden. Die Struktur (Spaltenanzahl und Spaltennamen) sowie der Ort

(Tabelle1) ist bei allen identisch. Auch wenn der Datenbereich der **Lebensmittel.xlsx** sowohl in Größe (Anzahl der Datensätze) als auch in der Position abweicht.

1. Schritt: **Quelle** = **Folder.Files(„Ordnerpfad“)**

Content	Name	Extension	Date accessed	Date modified	Date created	Attributes
Binary	Elektroartikel.xlsx	.xlsx	05.07.2025 12:51:49	04.07.2025 10:15:47	04.07.2025 10:12:32	Record
Binary	Haushaltswaren.xlsx	.xlsx	05.07.2025 12:51:51	04.07.2025 10:18:12	04.07.2025 10:16:34	Record
Binary	Lebensmittel.xlsx	.xlsx	05.07.2025 12:54:21	27.12.2022 17:46:50	04.07.2025 10:13:15	Record
Binary	Visualisierte Schrit...	.docx	05.07.2025 12:54:09	05.07.2025 12:54:08	05.07.2025 10:08:03	Record

2. Schritt: **NurExceldateien** = **Table.SelectRows(Quelle, each [Extension] = ".xlsx" or [Extension] = ".xls")**,
Nur Dateien mit der Endung .xlsx oder .xls zulassen.

Content	Name	Extension	Date accessed	Date modified	Date created	Attributes	Folder Path
Binary	Elektroartikel.xlsx	.xlsx	05.07.2025 09:27:50	04.07.2025 10:15:47	04.07.2025 10:12:32	Record	C:\Users\User\OneDrive\Dokumente\All_PQ\Dateien\Exceldateien\Einheitlich
Binary	Haushaltswaren.xlsx	.xlsx	04.07.2025 23:38:45	04.07.2025 10:18:12	04.07.2025 10:16:34	Record	C:\Users\User\OneDrive\Dokumente\All_PQ\Dateien\Exceldateien\Einheitlich
Binary	Lebensmittel.xlsx	.xlsx	04.07.2025 23:38:45	27.12.2022 17:46:50	04.07.2025 10:13:15	Record	C:\Users\User\OneDrive\Dokumente\All_PQ\Dateien\Exceldateien\Einheitlich

3. Schritt: **ContentZuTabelle** = **Table.FromList(NurExceldateien[Content], Splitter.SplitByNothing(), null, null, ExtraValues.Error),**
Die Liste der Spalte *Content* in eine Tabelle umwandeln

4. Schritt: Der wird etwas komplexer, da jede Zeile der neu zu erstellenden Spalte mehrere Bearbeitungsschritte benötigt.
Deshalb werden die **in eine eigene Abfrage gepackt**:
Die gesamte Anweisung des 4. Schrittes lautet:

```
AddTabellenspalte = Table.AddColumn(ContentZuTabelle,"Datei transformieren", each
    let
        Tab = Excel.Workbook([Column1], null, true),
        Tabelle1_Sheet = Tab{[Item="Tabelle1",Kind="Sheet"]}[Data],
        HeaderHoch = Table.PromoteHeaders(Tabelle1_Sheet, [PromoteAllScalars=true])
    in
        HeaderHoch
),
```

Es wird also eine Spalte Namens **Datei transformieren** erzeugt, die die Datentabellen der in **Column1** angegebenen Workbooks enthält.

The screenshot shows the Microsoft Power Query Editor interface. At the top, the formula bar contains the following DAX code:

```
= Table.AddColumn(ContentZuTabelle, "Datei transformieren", each
    let
        Tab = Excel.Workbook([Column1], null, true),
        Tabelle1_Sheet = Tab[[Item="Tabelle1", Kind="Sheet"]][Data],
        HeaderHoch = Table.PromoteHeaders(Tabelle1_Sheet, [PromoteAllScalars=true])
```

Below the formula bar, a data preview table is visible with two columns: 'Column1' and 'Datei transformieren'. The 'Column1' column contains three rows of data, all labeled 'Binary'. The 'Datei transformieren' column contains three rows of data, all labeled 'Table'.

On the right side, the 'Abfrageeinstellu...' pane is open, showing the 'EIGENSCHAFTEN' (Properties) and 'ANGEWENDETE SCHRITTE' (Applied Steps) sections. The 'ANGEWENDETE SCHRITTE' section lists the following steps:

- Quelle
- NurExceldateien
- ContentZuTabelle
- AddTabellenspalte** (highlighted in green)
- ...
- NurNötigespalten
- Spaltennamen
- TabellenExpandiere

A tooltip for the 'AddTabellenspalte' step is visible, stating: 'Schritt 4: Eine Spalte anhängen, die die Daten der in jeder Zeile angegebenen Dateien einliest und die jeweils 1. Zeile als Überschrift nimmt.'

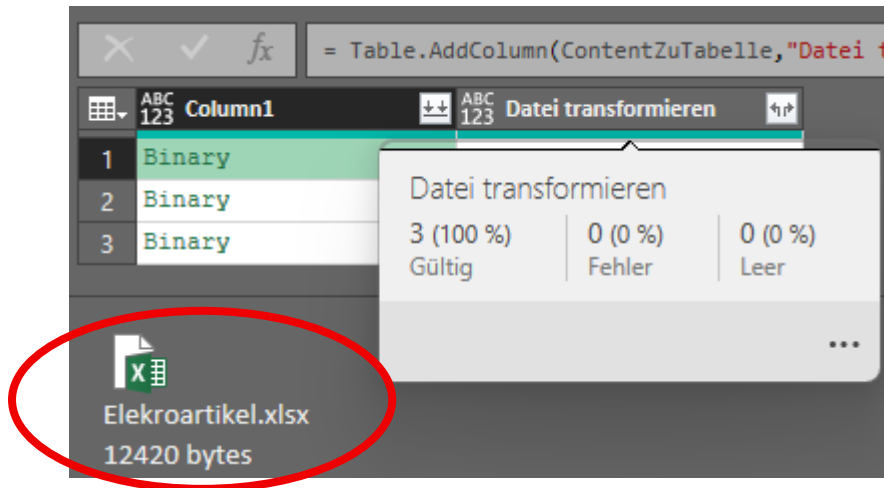
Der 3. und letzte Parameter der Table.AddColumn – Funktion erwartet eine Funktion zur Generierung des Spalteninhalts. Eine Funktion kann mit **each** oder einem Klammernpaar () => eingeleitet werden.

Eine Funktion als Ergebnis wäre für uns allerdings nicht hilfreich, da wir Tabellen für die spätere Auswertung benötigen.

Das Schlüsselwort each leitet dennoch unsere Absicht ein. Der Teil der separaten Abfrage ist der oben gelb hinterlegte. Schauen wir uns die nach **each** folgenden Einzelanweisungen etwas genauer an, die mit dem Schlüsselwort let eingeleitet werden.

Tab = Excel.Workbook([Column1], null, true),

Die Funktion **Excel.Workbook**(...) gibt eine Arbeitsmappe zurück. Wobei der 1. Parameter vom Typ **Binary** sein muss. Und genau das finden wir in der Spalte **Column1**. Wenn wir in den freien Bereich einer beliebige Zelle von **Column1** klicken, können wir unten in der Ausgabevorschau (hier rot umrandet) sehen, um welche Datei es sich handelt.



Die 2. Anweisung: **`Tabelle1_Sheet = Tab[[Item="Tabelle1";Kind="Sheet"]][Data]`**, weist das Programm an, dass die Daten aus allen Tabellenblättern (**Sheet**) mit dem Namen **Tabelle1** in der Spalte **Data** ausgelesen werden sollen.

Damit Du Dir das vorstellen kannst, schiebe ich mal die Anweisung dazwischen, mir die Daten des 1. Datensatzes (Achtung! Null-basiert) des Abfrageschrittes **ContentZuTabelle** anzeigen zu lassen. **Diese Anweisung ist NICHT Teil unserer Abfrage!**

Wie gut zu erkennen ist, gibt es dort eine Spalte Namens **Data**, deren Datenvorschau genau das liefert, was wir zu sehen wünschen. Bei der Gelegenheit will ich gleich darauf aufmerksam machen, dass (in diesem Fall) die eigentlichen Überschriften in Zeile 1 stehen. Das muss nicht immer der Fall sein, aber da es nun einmal so ist, werden wir im nächsten Schritt darauf reagieren.

✓

fx

= Excel.Workbook(ContentZuTabelle[Column1]{0}, null, true)

A^B

C

Name

Data

A^B

C

Item

A^B

C

Kind

✕

✓

Hidden

1

Tabelle1

Table

Tabelle1

Sheet

FALSE

Column1	Column2	Column3	Column4	Column5	Column6
ID	Produkt	Menge	Einheit	Preis	Bestand
1	Lampen	1	Stück	99	10
2	Stecker	1	Stück	0,8	12
3	Steckdosen	10	Stück	0,5	20
4	kabel 1,5	1	m	1,05	100

Denn der nächste und letzte Schritt unserer separaten Abfrage lautet:

HeaderHoch = Table.PromoteHeaders(Tabelle1_Sheet, [PromoteAllScalars=true])

...und bedeutet nichts anderes als die Werte der 1. Zeile als Überschriften zu verwenden....

Der letzte Schritt „**HeaderHoch**“ ist auch der, der in jede (each) Zeile der neuen Spalte ausgegeben werden soll.

- Schritt: Dieser Schritt ist nur deshalb notwendig, weil wir Werte zweier voneinander unabhängiger Tabellen miteinander in Verbindung bringen wollen. Das wären zum einen die Dateinamen, die wir aus Schritt 2 (nur Exceltabellen anzeigen) nehmen und in unser letztes Abfrageergebnis übertragen wollen. Das geht hier, weil die Tabellen der Spalte **Datei transformieren** noch nicht expandiert wurden. In beiden Fällen haben wir 3 Datensätze. Damit sie entsprechend zugeordnet werden können, müssen wir wissen, in welchen Zeilen sie stehen. Dazu benötigen wir kurzzeitig eine Indexspalte.

AddIndex = Table.AddIndexColumn(AddTabellenspalte, "Index", 0, 1, Int64.Type),

Hier geben wir an, dass die Spalte die Überschrift **Index** hat, mit **0** beginnt und in **Einer-Schritten** fortgeführt werden soll.

The screenshot displays the Power Query Editor interface. The formula bar at the top shows the expression: `= Table.AddIndexColumn(AddTabellenspalte, "Index", 0, 1, Int64.Type)`. The main workspace shows a table with three columns: 'Column1', 'Datei transformieren', and 'Index'. The 'Index' column has values 0, 1, and 2 for the three rows. On the right, the 'Abfrageeinstellungen' pane is open, showing the 'ANGEWENDETE SCHRITTE' (Applied Steps) list. The steps are: 'Quelle', 'Nur Exceldateien', 'ContentZuTabelle', 'AddTabellenspalte', 'AddIndex', 'Dynamenspalte', 'Spalten', 'Spaltennamen', and 'TabellenExpandieren'. The 'AddIndex' step is currently selected and highlighted in green. A tooltip for this step is visible, containing the text: 'AddIndex' and 'Schritt 5: Eine Indexspalte hinzufügen, damit die Dateinamen korrekt zugewiesen werden können'.

6. Schritt: **AddDateinamensspalte** = **Table.AddColumn(AddIndex, "Dateiname", each NurExceldateien[Name][[Index]])**, Entscheidend ist der gelb unterlegte Teil. Sieh im Schritt **NurExceldateien** in der Spalte **[Name]** in der **{Zeile nach, die in der Spalte [Index]}** steht. Wobei die Zeilenangabe immer in **{ }** und die Spaltenangabe in **[]** steht. Die Zeile, die Spalte Index steht ist somit:

{[Index]}

Eine direkte Zeilenangabe für Zeile 4 wäre: **{3}**

Abfrageeinstellungen

ANGEWENDETE SCHRITTE

- Quelle
- NurExceldateien
- ContentZuTabelle
- AddTabellenspalte
- AddIndex
- AddDateinamensspalte**
- NurNötigeSpalten
- ...
- ...

AddDateinamensspalte
Schritt 6: Hier wird jetzt die Spalte mit den korrekt zugewiesenen Dateinamen aus der Spalte "Name" der Abfrage "NurExceldateien" angehängt

7. Schritt: **NurNötigeSpalten** = **Table.RemoveColumns(AddDateinamensspalte,{"Column1","Index"})**, Schnell erklärt: Alle nicht benötigten Spalten (in diesem Fall **Column1** und **Index**). Nicht benötigte Spalten (auch wenn es nur eine ist) werden als Liste in **{ }** und Anführungszeichen angegeben. Sind es mindestens 2 Spalten, werden sie kommasepariert,

Formula bar: `= Table.RemoveColumns(AddDateinamensspalte,{"Column1","Index"})`

ABC 123	Datei transformieren	ABC 123	Dateiname
1	Table		Elektroartikel.xlsx
2	Table		Haushaltswaren.xlsx
3	Table		Lebensmittel.xlsx

Abfrageeinstellungen

EIGENSCHAFTEN

Name: OhneHilfsprogramme (2)

Alle Eigenschaften

ANGEWENDETE SCHRITTE

- Quelle
- NurExceldateien
- ContentZuTabelle
- AddTabellenspalte
- AddIndex
- AddDateinamensspalte
- NurNötigeSpalten**
- Spaltennamen
- ...

NurNötigeSpalten
Schritt 7: Unnötige Spalten entfernen

8. Schritt: **Spaltennamen** = `Table.ColumnNames(AddIndex[Datei transformieren]{0})`,

Diesen Schritt fügen wir deshalb ein, um dynamischer auf Änderungen im nachfolgenden Schritt reagieren zu können. Egal, ob der Pfad geändert wird, oder sich die Spaltenbezeichnungen und/oder deren Anzahl ändert, es wird immer eine Liste mit den Überschriften geben, die **im 1. Datensatz** (nullbasiert) des Abfrageschrittes **AddIndex** in der Spalte **Datei transformieren** zu finden sind. Und das wären diese:

Formula bar: `= Table.AddIndexColumn(AddTabellenspalte, "Index", 0, 1, Int64.Type)`

ABC 123	Column1	ABC 123	Datei transformieren	ABC 123	Index
1	Binary		Table		0
2	Binary		Table		1
3	Binary		Table		2

ID	Produkt	Menge	Einheit	Preis	Bestand
1	Lampen	1	Stück	99	10
2	Stecker	1	Stück	0,8	12
3	Steckdosen	10	Stück	0,5	20
4	kabel 1,5	1	m	1,05	100

Abfrageeinstellungen

EIGENSCHAFTEN

Name: OhneHilfsprogramme (2)

Alle Eigenschaften

ANGEWENDETE SCHRITTE

- Quelle
- NurExceldateien
- ContentZuTabelle
- AddTabellenspalte
- AddIndex**

Zum Vergleich – Das eigentliche Abfrageergebnis:

fx = Table.ColumnNames(AddIndex[Datei transformieren]{0})

Liste
1 ID
2 Produkt
3 Menge
4 Einheit
5 Preis
6 Bestand

Abfrageeinstellungen

EIGENSCHAFTEN

Name
OhneHilfsprogramme (2)

Alle Eigenschaften

ANGEWENDETE SCHRITTE

- Quelle
- NurExceldateien
- ContentZuTabelle
- AddTabellenspalte
- AddIndex
- AddDateinamensspalte
- NurNötigeSpalten
- Spaltennamen**
- TabellenExpandieren

9. Schritt: **TabellenExpandieren** = **Table.ExpandTableColumn(NurNötigeSpalten, "Datei transformieren", Spaltennamen, Spaltennamen)**

Zu guter Letzt werden die in der Spalte **Datei transformieren** der Abfrage **NurNötigeSpalten** enthaltenen Tabellen expandiert. Dabei wird die im Vorgängerschritt ermittelte Liste der Spaltennamen für die letzten beiden Parameter der Funktion **Table.ExpandTableColumn** verwendet.

fx = Table.ExpandTableColumn(NurNötigeSpalten, "Datei transformieren", Spaltennamen, Spaltennamen)

ID	Produkt	Menge	Einheit	Preis	Bestand	Dateiname
1	Lampen	1	Stück	99	10	Elektroartikel.xlsx
2	Stecker	1	Stück	0,8	12	Elektroartikel.xlsx
3	Steckdosen	10	Stück	0,5	20	Elektroartikel.xlsx
4	kabel 1,5	1	m	1,05	100	Elektroartikel.xlsx
5	Besen	1	Stück	99	10	Haushaltswaren.xlsx
6	Schaufel	1	Stück	0,8	12	Haushaltswaren.xlsx
7	Staublappen	10	Stück	0,5	20	Haushaltswaren.xlsx
8	Wäscheleine	1	m	1,05	100	Haushaltswaren.xlsx
9	Mehl	1	kg	0,99	10	Lebensmittel.xlsx
10	Zucker	1	kg	0,8	12	Lebensmittel.xlsx
11	Salz	500	g	0,5	20	Lebensmittel.xlsx
12	Milch	1	l	1,05	5	Lebensmittel.xlsx
13	Butter	1	Stück	1,5	20	Lebensmittel.xlsx

Abfrageeinstellungen

EIGENSCHAFTEN

Name
OhneHilfsprogramme

Alle Eigenschaften

ANGEWENDETE SCHRITTE

- Quelle
- NurExceldateien
- ContentZuTabelle
- AddTabellenspalte
- AddIndex
- AddDateinamensspalte
- NurNötigeSpalten
- Spaltennamen
- TabellenExpandieren**